September/October 1991 Edition

RERUN

RUN Programs on Disk

For the C-64 and C-128

Plus: Extra Bonus Programs!

Introduction

September/October 1991 ReRUN

THIS SEPTEMBER/OCTOBER EDITION of ReRUN marks another significant milestone in our long and illustrious history. After this edition, the documentation that you normally read here will become an integral part of the disk itself. That means that a separate booklet will no longer be necessary.

Don't worry—these lively editorials will still be there, only on disk rather than on paper. Of course, if you prefer to read from a printout, we're also providing the option to print all the documentation found on the new disk. Thanks to a new ReRUN interface, every new edition will be equipped with documentation on disk, an interactive design, and an elimination of the current C-64 and C-128 menu systems.

While we've always prided ourselves on Basic listings that users could examine and learn from, we'll still provide the RUN listings in the same form that they appear in the magazine so that they can be listed and examined. With this new interface, we look forward to publishing ReRUN for years to come.

The designer and programmer of the new user interface for ReRUN, Robert Rockefeller, also wrote the premier program of the September/October issue of RUN, **MultiCopy 64/128**. This program is one of those essential disk utilities that no self-respecting Commodore user should be without.

While most utilities offer the usual file copy and disk maintenance commands, MultiCopy offers niceties that other copy programs lack. It supports up to three disk drives, can date and time-stamp files, and offers a Verify option. This option is useful and unique in that it can verify files after copying, ensuring that no file integrity is lost.

Tony Brantner is our reigning champ of 64 machine language games, and he's back with **Rollerdash**. This program lets you skate down an obstacle-strewn boulevard, jumping and dodging all manner of items designed to slow your progress.

Fraction Action is the next program on the list. The object here is to race about a grid on the screen identifying and selecting only those fractions that appear in their lowest form. A dangerous creature roams about the screen, just to keep things exciting.

Classy Graphics, the next on the list, also hails from the Septem-

ber/October issue of RUN. It allows you to use Animation Station graphics as backdrops in your C-128 programs. In order to dissipate any reservations you may have towards the usefulness of this utility, we've really pulled out the stops with demos created with the program.

First, we put *Halloween Treat* on the disk, an Animation Station pumpkin with sprites added for animated effects. Put this on display this Halloween and watch the kids' delight. Also, there's *Stopwatch*, the default file on this disk, which displays the beautiful detail that Animation Station graphics bring to your 128's 40-column screen.

Mark Jordan's **128 Mode** program this month is actually a C-64 program. Keypad In 64 is just that—a program that activates the C-128's keypad in 64 mode. It also awakens the otherwise dormant cursor keys at the top of the C-128's keyboard.

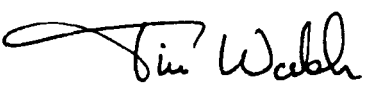
Bonus programs are in good supply this time. **Scramble** is a C-64 mode program that allows up to six players to compete in a race to make words from scrambled letters on the screen.

Menu Runner is another C-64 mode program containing three subprograms. It works similar to the principle of the new user interface we'll begin using in our next ReRUN edition. You run this menu program, and whenever you want to access the menu again, just press F7 from within any of the sub-programs.

The first sub-program is *Moon Phases*. This shows you the phase of the moon on a given date and time. It's a great way to find the moon's phase at the time of your birth. *Elapsed Time*, the next sub-program, shows the number of years, months, days, and hours between two days in history. Find out how many "days old" you are—to the current hour! The third sub-program, *Number Conversion*, converts between decimal, binary, hexadecimal and more.

For C-128 fans, we've included William Wright's **Menu Maker**, an easy-to-use menu creator program. Finally, there's **Sourcemaster**, a commercial-quality program for creating assembly language files from machine language files on disk. Budding and experienced programmers alike will find this a useful utility to add to their collection.

That's it for this edition of ReRUN. I'll be back soon, in a new, improved format. Stay tuned—you won't be disappointed.



Tim Walsh
Technical Manager

Directory

PAGE	DOCUMENTATION	DISK FILENAME	FILE TYPE
		MENU 128	BASIC
		MENU 64	BASIC
1	MULTICOPY (C-64 & 128)	MULTICOPY.HEX	BASIC
		MULTI-COPY	ML
5	ROLLERDASH (C-64)	ROLLERDASH.HEX	BASIC
		ROLLERDASH	ML
6	FRACTION ACTION (C-64)	FRACTION ACTION	BASIC
7	CLASSY GRAPHICS (C-128)	HALLOWEEN TREAT	BASIC
		SPLIT GRAPHIC	BASIC
		GHOST1.SPRITE	ML
		PUMPKIN1.SPRITE	ML
		SCRN.PUMPKIN1	ML
		VMTX.PUMPKIN1	ML
		CRAM.PUMPKIN1	ML
		PI.COLORWATCH	ML
		SCRN.N.PUMPKIN1	ML
		VMTX.N.PUMPKIN1	ML
		CRAM.N.PUMPKIN1	ML
		SCRN.COLORWATCH	ML
		VMTX.COLORWATCH	ML
		CRAM.COLORWATCH	ML
8	128 MODE (64 MODE)	KEYPAD IN 64	BASIC
		LOADER	BASIC
		KEYPAD.ML	ML
9	SCRAMBLE (C-64)	SCRAMBLE	BASIC
10	MENU RUNNER (C-64)	MENU RUNNER	BASIC
		MOON PHASES	BASIC
		ELAPSED TIME	BASIC
		NUMBER CONVERT	BASIC
13	MENU MAKER (C-128)	MENU MAKER	BASIC
14	SOURCEMASTER (C-64)	SOURCEMASTER.ML	ML

Before you run a program, carefully read the documentation that pertains to it.

How to Load

Loading from the Menu

To get started, C-64 users should type LOAD "MENU 64",8 and press the RETURN key. When you see the Ready prompt, the menu is loaded; simply type RUN to see a list of the programs on your disk. C-128 users need only press the SHIFT and RUN-STOP keys. When the program titles are displayed on the screen, you can run the one you select by pressing a single key.

Loading from the Keyboard

If you prefer not to use the menu program, follow these instructions:

C-64: To load a C-64 program written in Basic, type: LOAD"DISK FILENAME",8 and then press the RETURN key. The drive will whir while the screen prints LOADING and then READY with a flashing cursor beneath. Next type RUN and press the RETURN key. The program will then start running. Type LOAD"DISK FILENAME",8,1 to load a C-64 program that is written in machine language (ML).

C-128: All C-64 programs can be run on the C-128 as long as your computer is in 64 mode. All C-128 programs are clearly labeled on the directory page. Your C-128 *must* be in 128 mode to run these programs. To load a 128 mode program, press the F2 key, type the filename and then press the RETURN key. Type RUN when the program has loaded.

Making Copies of ReRUN Files

Many programs on your ReRUN disk have routines that require a separate disk onto which the program writes or saves subfiles. To use these programs, you must first make a copy of the original program on another disk that has enough free space on it to hold these newly written subfiles.

It's simple to make a copy of a Basic program. Just load it into your computer as outlined above, and then save the program back onto a separate disk that has plenty of free space for extra files.

Copying an ML program is not so simple. You cannot just load and save it; you'll need to use a disk-backup utility program, such as the one on your Commodore Test Demo disk.

RUN it right: **C-64 or C-128 (in 40- or 80-column mode)**

MultiCopy

By Robert Rockefeller

IT'S A BREEZE duplicating Commodore files with MultiCopy. It works with a C-64 or a 128 in 40- or 80-Column mode. It accommodates one, two or three disk drives, and provides 11 commands, including Copy, Verify, Scratch, Rename and Format. It also time- and date-stamps files.

Select the first option (a) from either Menu 64 or Menu 128 to launch MultiCopy. At the beginning of the program, you must enter the date in mm/dd/yy format: month, day and year. The month must be represented as a number from 01 to 12, the day from 01 to 31, and the year from 00 to 99 (each entry must be a two digit number).

The program checks to see if the 24-hour clock has already been set in a previous MultiCopy session. If the clock isn't running, you must enter the time in hh:mm:ss format, which means hour, minute and second, respectively. As with the date, each entry must be made as a double digit, and remember that this is a 24-hour clock, so enter the hour accordingly. (The clock and the date are not linked, so the date doesn't change when the clock runs to midnight.)

The Main Screen

After you have entered the date and time information, the Multi Copy main screen appears, containing three boxes titled Current Devices, Commands, and Settings. A fourth box displays the time and date.

MultiCopy checks for disk devices each time it starts up, and automatically determines the type and device number of each one connected. The Current Devices box lists the disk devices connected and the device number associated with each. MultiCopy supports the 1541, 1571 and 1581 disk drives, and their compatibles, as well as the 1750 RAMdisk.

The Settings box lists the current program settings—from device indicates the device to load from when copying (or verifying) files; to device indicates the device to copy files to. These can be changed with

the Usage command. Prompt can be turned on or off. When it's on, the program warns you before copying over any existing file. The program defaults to the off setting. You can change the setting with the Prompt command. (For details about this setting, see the Copy Files section below.)

The Commands box lists the available commands. Use the cursor keys to move the cursor up and down until it's positioned on the desired command, then press the RETURN key to execute. If you select a command and then change your mind, simply press the run-stop key to abort.

THE COMMANDS

•**Copy Files**—Select this command to copy files from one disk to another, using either one or two disk devices. The files will be loaded from and written to the devices you selected in the Settings box on the main screen. If there are no files on the from disk, no directory will be displayed. If the from and to devices are the same, you must swap disks periodically.

When you're copying a file, a file with the same name could already exist on the destination disk. A word of caution: Unless the prompt is on, the program will automatically scratch the old file, writing over it with the new one.

When you press RETURN to execute the Copy command, a box will appear listing all the files on the disk in the from device. With the cursor keys, position the cursor on a file you wish to copy, then press the space bar to tag the file with a checkmark. Select as many files as you need.

The BACK-ARROW key (in the upper-left corner of the keyboard) performs a special function. When it is pressed, any unmarked files become marked and any marked files become unmarked. For instance, let's say you want to copy all the files on a disk except one. You can tag the one you don't want, then press the BACK-ARROW key to reverse the operation. The computer will untag the one that you don't want and tag all of the others.

Press RETURN to copy the checked files. During the load operation, the cursor will indicate which file is currently in transit. When the program has loaded as many files as it can, it writes them to the destination disk. If you're using one drive, a prompt will tell you to insert the to disk. Just remove the from disk, insert the to disk and press RETURN.

This process of loading and writing files will continue until all of the tagged files have been copied. Then MultiCopy will read in the directory blocks of the to disk and stamp each file that was copied with the date and time.

Because Commodore didn't make the 1750 RAMDOS software 100-percent compatible with Commodore floppy drives such as the 1541, it's impossible to copy a relative file from a RAMdisk to a floppy. Time- and date-stamping isn't possible with RAMdisks either.

•**Prompt On/Off**—Select this command to turn the prompt on or off. If you wish to be notified before files are scratched during copy operations, press Y. If you don't, press N and RETURN.

•**Verify Files**—Select this command to verify files on different disks. With two disk devices, files will be loaded from the from device and compared to a file in the to device. If you're using only one device, you have to swap disks before the comparison can take place.

Press RETURN and a box will appear listing all the files on the disk in the from device. Use the cursor keys to place the cursor on the file you wish to verify, then press the space bar or the RETURN key to select that file. A checkmark will appear beside the filename, and the file will be loaded into memory.

When the file has loaded, you will be asked to select which file on the destination disk to compare it to. If you're using one drive, a prompt will tell you to insert the to disk before verification can occur. Just swap the disks and press RETURN. If you're using two drives, no prompt will appear and the files will be compared automatically. If the files are identical, a message will appear saying FILES OK! Otherwise the message will say FILES DON'T VERIFY! In either case, press RETURN to revert to the Main screen.

•**Scratch Files**—Select this command and enter the device number. A box will appear listing all the files in that device. Place the cursor on each file you wish to scratch, then press the space bar to tag the file. The BACK-ARROW key performs the same function here as with the Copy command.

•**Rename Files**—Select this command and enter the device number you wish to use, and a directory will appear. Select the file you wish to rename by cursoring to it and pressing the space bar. When a prompt asks RENAME FILE TO WHAT?, enter the new filename and press

RETURN. After you've marked the files you want, press RETURN and they will be renamed.

•**Usage**—The Usage command lets you define the from and to devices for copy and verify operations. Files will be loaded from the from device and written to or compared with a file in the to device. Just enter the appropriate device numbers at the prompts as they appear. The from and to devices can be the same or different drives.

•**Format Disk**—To format a disk, enter the device number and press RETURN at the first prompt. At the second prompt, enter the disk name and press RETURN. At the third prompt, enter the disk ID, which must be two characters long, and press RETURN. The disk will then be formatted.

•**Erase Disk**—The Erase command lets you scratch the contents of a disk. At the first prompt, enter the disk device number and press RETURN. At the second prompt, enter the new disk name and press RETURN. All the files on the disk will be scratched.

•**Poll Devices**—This command determines which disk devices are connected to the computer and displays them in the Current Devices box. MultiCopy can recognize up to three devices and automatically checks for them when starting up. The Poll Devices command is provided in case you turn on another disk device while the program is running. Note that you may have to issue a new Usage command after Poll Devices executes, because the latter automatically resets the from and to devices.

•**Compatible**—With this command you can define a third-party, non-Commodore disk drive as compatible with a 1541, 1571, or 1581. You must make sure that the drive is indeed compatible—for example, a 1581 isn't compatible with a 1541 or 1571. At the prompts asking DEFINE WHICH DEVICE AS COMPATIBLE?, enter the device number of the compatible drive and press RETURN. At the next prompt, asking COMPATIBLE WITH WHICH DRIVE?, enter the drive with which it's compatible—either a 1541 or 1571—then press RETURN.

You can copy a relative file only from a 1541, 1571 or 1581, although relative files can be written to other types of drives. Also, time- and date-stamping is possible only with Commodore floppy drives and their compatibles, because the directory track must be directly accessed for these functions.

•**Quit**—Select this command when you wish to stop using the program and return to Basic.

With MultiCopy you can copy files quickly and efficiently. All the commands you need are here, ready for you to put them to work.

RUN it right: **C-64; joystick**

Rollerdash

By Tony Brantner

SLIP ON YOUR BLADES and slide into your spandex—you'll need to be agile to make it through this challenge! In Rollerdash, you race down a sidewalk strewn with trashcans and other obstacles.

Load Rollerdash using Menu 64 then move your joystick in any direction to begin the action. Each time you tap the joystick to the right, the skater will push forward, accompanied by sound effects. Keep him pushing to build up speed; let him relax and he'll slow down again.

When the skater approaches a tire, garbage can or barricade, press the joystick up to make him leap. He can make it if he's built up enough speed and you time the jump correctly. He'll also meet low signs hanging out from the brick wall. Pull the joystick down to make him duck. If he bumps into an obstacle, he'll slow down and have to regain his momentum.

Current speed, distance traveled and time remaining are displayed at the bottom of the screen. Each time the distance read-out increases, your score rises ten points. Reach the end of the sidewalk, and you get a bonus of 100 times the seconds left on the clock. At level 1, the timer starts counting down from 100; it decreases by five at each new level. Any time you want to pause the game, press the SHIFT-LOCK key.

To play the game, load and run the program, making sure your joystick is plugged into port 2. You'll soon be up to speed.

Fraction Action

By William Snow

THE MOST DIFFICULT PART of working with fractions, whether adding, subtracting, multiplying or dividing, is making sure the answer is in simplest form—that is, with no divisor common to both the numerator (the number above the line) and the denominator (the number below the line). For example, $^{1}/_{3}$ and $^{5}/_{17}$ are in simplest form, but $^{2}/_{6}$ and $^{4}/_{16}$ are not (they're reducable to $^{1}/_{3}$ and $^{1}/_{4}$, respectively).

I wrote Keep It Simple (K.I.S.) to help my fifth-grade students distinguish between fractions that are simple and those that are not, and it gives them all the practice they need. The game screen is a colorful five-by-five grid, with a randomly chosen fraction in each square. Numerators and denominators can be up to 19, and play is accompanied by appropriate sound.

Load the program using Menu 64. The game seems easy at first: All you do is use the joystick to move the smiley face around the grid and hit the fire button on fractions that are simple. But then you find you mustn't fire on fractions that aren't simple or on blank spaces. And watch out for that simpleton! If you take too long thinking about a fraction, he'll catch up with Smiley and you'll lose a life. Lose three lives and the game is over. To add challenge, the simpleton's speed increases as the game progresses. When the game is over you can play again or quit—or you may just want to study the fractions you missed for a while.

Because playing K.I.S. is so much fun, students may not even be aware that (Heaven forbid!) they're learning. It's surprising how quickly they *will* learn; after getting zapped a few times, they'll be checking fractions quickly and accurately. What could be . . . *simpler*?

Classy Graphics

By Michael Falco

THE C-128 HAS MANY fine features, not the least being Basic 7.0 with its powerful graphics commands. To complement these commands and to create background pictures for my C-128 programs, I wanted to use my Animation Station software and tablet. They make drawing on the computer screen as easy as drawing on paper.

Because I couldn't find a utility program that would convert the Animation Station graphics to Basic 7.0 format, I wrote Split Graphics to fill the void. It can be used for any kind of graphics your programs might need.

To use Split Graphics, first create and save a graphics image using the standard Animation Station procedures. Then return to 128 mode on your computer and run the program. At the prompt, type in the complete filename of your graphics image (including the PI. prefix) and press the RETURN key.

Split Graphics will create three separate files from the single original and give them three new prefixes: SCRN, VMTX and CRAM. The first file contains 8K of screen memory, the second 1K of video matrix, and the third 1K of color RAM. The program will load these files into appropriate memory areas for display and verification. When you enter a carriage return, it will return you to normal Text mode.

Programmers wanting to use their own Animation Station graphics in a Basic 7.0 program can BLoad the three files into the appropriate areas of memory and initialize the various background graphics modes using the commands listed within Split Graphics.

Just in time for Halloween, there's a program on this disk done with Split Graphics called *Halloween Treat.* It was created with Animation Station. Select it using Menu 128 and you'll be the talk of the neighborhood this October 31.

Now, with the quality of Animation Station graphics available in your Basic 7.0 programs, you can give them a professional touch of class.

128 Mode

By Mark Jordan

RECENTLY I DID something highly distasteful with my C-128: I went over to 64 mode to do some programming. I was working on a machine language project, so I really didn't mind leaving Basic 7.0 commands behind. I wasn't even that bothered by the lack of editing commands in 64 mode, because I have a utility that gives me some of them back.

What really hurt was leaving behind my extended keyboard. I needed to type in a lot of numbers, and, oh what agony to tap on the C-128's numeric keypad and watch it play dumb. I decided it was time to do something about it.

What I Did About It

This program, that I named "Keypad," is a short machine language routine that awakens both the keypad and the top-row cursor keys in 64 mode on the 128. Plus, it gives most of the other mute keys in the top row something to say.

To activate it, just select KEYPAD IN 64 using Menu 64.

Besides bringing the keypad and cursor keys to life, this program gives the following keys new identities:

ESCAPE—toggles Quote mode on and off.
TAB—jumps the cursor eight spaces to the right.
ALT—toggles between Upper- and Lowercase mode more easily than SHIFT/COMMODORE does.
HELP—clears the screen from the cursor downward.
LINE FEED—jumps the cursor to the bottom of screen.
NO SCROLL—pauses listings (or anything else). Just press any other key to continue.

Technicalities

If you have a technical bent and are wondering how Keypad works, the key ingredient is a new register in the VIC (8564) chip that communicates with memory location 53295 ($D02F). On the original C-

64, this location wasn't connected to the VIC-II chip, so reading or writing to it was useless. Now it's connected for the sole purpose of reading the additional keys on the C-128's keyboard.

The reason the Commodore engineers didn't implement this register in 64 mode from the start is because they were trying to create a virtual 64. That's a pity, though, because the register doesn't affect C-64 programs at all. I suppose it's possible that some C-64 software was written that tapped into location 53295 and would thus be corrupted, but it's unlikely.

Programming in the 64 mode of a C-128 can be aggravating for the experienced 128 programmer. But, it has its benefits and advantages. I hope my Keypad program will encourage other diehard 128 programmers to take the plunge and give 64 mode a try.

RUN it right: **C-64**

Scramble

By R. B. Cook

SCRAMBLE CHALLENGES up to 12 players to create words from a five-by-five matrix of letters. Start by selecting Scramble using Menu 64, then enter each player's name. You can edit a name by pressing RETURN until the cursor is on it. When you're ready to play, press the UP-ARROW key to scramble the letters and start the timer.

You have three minutes to write down as many words as possible. The words must be at least three letters long, they can't be proper nouns, and they may connect horizontally, vertically or diagonally. You may use each letter only once within a word.

Here's an example:

A E T
Z S A

You can create TEA, TEAS, SEAT, SET, SAT, ATE, EAT and EATS from these six letters. SEATS isn't legal, because the S may be used only once. TEAS can be created two ways, using each A, but counts only once.

Since Q almost always appears with U, they're combined in one square but count as two letters.

When the time is up, all the players must compare their lists for duplicate words. If a word is found on more than one list, it must be crossed off all the lists. Score any remaining words according to the values in the red box at the top of the screen, then enter the points to the right of the players' names. After all the points have been entered, press ESCAPE to add the points to the scores and reorder the players' names in descending order of score. As you can see, unusual words will help you win at Scramble.

Press UP-ARROW to scramble the letters for the next round or BACK-ARROW to start a new game.

RUN it right: **C-64**

Menu Runner

By Terry Burris

HERE'S A NIFTY little gem of a program that calculates elapsed time intervals, moon phases and a number of useful number-base and unit conversions. Each function is actually a separate routine that's called from the main menu.

Main Menu

At the main menu, select the routine you want by number. You can return from a routine to the main menu at almost any time by pressing F7, or terminate the entire program by pressing F8.

Each routine will ask you for input of some type. Press RETURN after making each entry. If you change your mind about an individual entry, clear it by pressing delete before RETURN. To clear all entries, press HOME before RETURN for the last entry.

Moon Phases

This routine requests the year, month, date and hour. The year must be a four-digit number from 1582 to 2500, and the month can be any number from 1 to 12. Entries for the date are limited to the number of days in the particular month. The routine will accept 29 days for February only if it's a leap year. The hour operates on a 12-

hour clock, so AM or PM must be designated. Don't skip any entries.

After you enter the hour, there'll be a delay as the computer processes the data, then the screen will show the name of the moon phase on the date you specified, the day of the week, and a large graphics representation of the moon. Any holiday on that date will also be noted. After five seconds you can press any key to clear the screen for another entry.

Elapsed Time

This routine determines the interval between any two points in time. The rules concerning entries are the same as for Moon Phases. Elapsed Time requests that you enter the dates to calculate to and from. Then the program shows the approximate interval—expressed in years, months and tenths of months—in the lower-left portion of the screen. The interval is expressed in more detail, with days and hours, in the lower-right area of your screen.

Number Conversions

This routine converts numbers between various bases and units. Select conversions by number, and press M to return to the routine's menu. You can switch between conversions in a complementary pair by pressing S, but the complement you switch from will be cleared. To clear the screen of all numbers, switch to the complement then back again.

- *Decimal to Binary:* Enter any decimal number from 1 to 65,535.
- *Binary to Decimal:* First enter, in decimal, the number of digits in the binary number, to a maximum of 16. Then enter the actual binary digits.
- *Decimal to Hexadecimal:* Enter any decimal number from 1 to 65,535.
- *Hexadecimal to Decimal:* First enter the number of digits in the hexadecimal number, to a maximum of four. Then enter the hexadecimal digits.
- *Rectangular to Polar Coordinates:* Enter X and Y as any numbers up to seven digits long in the range –999.99 to 999.99. The answer will be returned as a distance and an angle.
- *Polar to Rectangular Coordinates:* For the distance (termed the modulus), enter a number up to seven digits long between 0 and 1000. Enter an angle between 0 and 360 degrees. The answer will be returned in electronic notation as "real" and "imaginary" parts of the vector, corresponding to X and Y.

ENGLISH TO METRIC

This option offers five different conversions. On selection, it starts with miles to kilometers. Step through the others by pressing RETURN without an entry. As long as you enter numbers, the same conversion will keep coming up.

•*Miles to Kilometers:* Enter any number up to eight digits long in the range 0 to 62,137.11.

•*Feet and Inches to Meters:* For feet, enter any number up to seven digits long between 0 and 5280. For inches, enter any number up to five digits long between 0 and 12. You can also enter feet as a decimal number, then skip inches by just pressing RETURN.

•*Square Feet to Square Meters:* Enter any number up to eight digits long that's between 0 and 100,000.

•*Gallons to Liters:* Enter any number up to eight digits long between 0 and 26,415.58.

•*Pounds and Ounces to Kilograms:* For pounds, enter any number up to eight digits long that's greater than 0 and less than 99,999. For ounces, enter any number up to four digits long between 0 and 16. You can also enter pounds as a decimal number, then skip ounces by just pressing RETURN.

METRIC TO ENGLISH

These options invert the previous five conversions.

•*Kilometers to Miles:* Enter any number up to eight digits long between 0 and 100,000.

•*Meters to Feet and Inches:* Enter any number up to seven digits long between 0 and 1000.

•*Square Meters to Square Feet:* Enter any number up to eight digits long that's between 0 and 9290.301.

•*Liters to Gallons:* Enter any number up to eight digits long that's between 0 and 100.

•*Kilograms to Pounds and Ounces:* Enter any number up to eight digits long that's between 0 and 45,325.78.

Menu Maker

By William P. Wright

TIRED OF TRYING TO find programs in disk directories? Or trying to stop the screen scroll before programs race out of sight? Here's a menu program for the C-128 in either 40- or 80-Column mode that solves these problems and is far easier to use than the disk directory.

Just for display purposes, the files from this edition of *ReRUN* are listed in Menu Maker. You can substitute your own, beginning at line 1000. First, decide which programs you want to have on a disk. Then add the program names as Data statements, starting at line 1000. The last program name must be followed by the word end. Now save Menu Maker with filename menu at the beginning of the disk, and follow it with the program files. As you acquire more programs for your disk, add their names to the Menu Maker Data statements and resave the program.

To operate Menu Maker, just press SHIFT-RUN/STOP. Then move the highlight to the program you want to run and press RETURN.

If you want to run 40-column programs with either a 40- or 80-column display, add the following line to Menu Maker:

```
500 IF RGR(0)=5 THEN PRINTCHR$(27)"X":SLOW
```

If you want to run 80-column programs with either a 40- or 80-column display, add this line:

```
500 IF RGR(0)<>5 THEN PRINTCHR$(27)"X":FAST
```

Saving Menu Maker as MENU on all disks makes continuous directory scanning possible.

This program creates a menu with the same basic structure as those in commercial software. Don't hesitate to use modified portions of it in your programs. I'm sure you'll be pleased with the results.

Sourcemaster

By Mike Gregory

IF YOU'RE SERIOUS about writing programs in machine code, it's more or less mandatory that you use an "assembler," such as the one in the Commodore Development Package. An assembler transforms "pseudo-code," in the form of a sequential file, into executable machine code. The pseudo code, also called "source code," can be understood easily by you; the machine code is understood by your C-64.

Source code makes it easy to study and modify the inner workings of a program. Adding another step or changing the destination of a branch instruction is just a minor amendment on your part, then assembly produces a complete, new program.

The source code for programs you've written yourself is, of course, available. The source code for programs you've typed in from a magazine or gotten from the public domain usually is not. To study or modify such a program you need an "unassembler," which converts machine code back into source code in a form that can be edited and then reassembled. Sourcemaster is an unassembler that complements the assembler in the Commodore Development Package.

How Does Sourcemaster Work?

The easiest way to explain what Sourcemaster does is to describe what happens during each "pass" of the unassembly process. While this description relates specifically to Sourcemaster, any unassembler must include the same general processes.

Just select Sourcemaster using Menu 64, then put a workdisk in the drive that contains a machine language file. In the first pass, you tell Sourcemaster which file to work on—that is, the filename of the machine language file. It then reads the file and determines the start and finish addresses.

The second pass displays the two addresses and asks you to supply a filename for Sourcemaster to give the source code it will produce. The default, selected by pressing RETURN, is the name of the original machine code file with an .A suffix. Pass two then asks for the start and

end addresses of the portion of machine code you want to unassemble. At the prompt for an unassembly start address, you may select any value within the machine code file. The default is the start of the file. The entry routine lets you enter either decimal or hexadecimal numbers. A hex entry must be preceded by a $ symbol.

Sourcemaster will then ask you to designate any noncode areas within the unassembly. Don't be intimidated by this—it just helps you avoid spurious code in the source listing that could come from text, byte tables and such in the machine code. If you want to indicate such regions—and I recommend it—you need to have first used the .D command of a machine code monitor, such as the one in the Development Package, to do a preliminary scan of the machine code to detect any byte or text tables. When you enter table addresses, Sourcemaster will continue to prompt for more, so a series of separated tables can be entered. Pressing RETURN after entering the last table will get you out of the loop. Whether or not you indicate such noncode regions, reassembly of the source code will still produce the original machine code.

Finally, you'll be asked for the unassembly finish address. The default value is the end of the machine code file. In many cases, particularly for short pieces of code, you need only accept the defaults to complete the second pass.

In pass three, Sourcemaster reads through the section of machine code you've selected for unassembly and collects all the addresses used within it. These may be either absolute or zero-page addresses, and the table produced will form the basis for the symbols and labels used within the source code file. The addresses are simply collected as they occur to speed up the pass.

Pass four sorts the addresses into ascending order, using a "quicksort" algorithm that's been optimized for the 6502 microprocessor. This sort is very fast; with machine code of only a few blocks you'll have to watch closely to even see the pause as the pass number flashes on the screen! Once the addresses are sorted, any repeated values are removed. The outcome is a table of addresses in ascending order.

Pass five is actually a multiple pass that produces a series of source code files. The repeated passes are necessary because of memory size constraints. The Development Package editor can handle only individual source code files that fit into Basic RAM. These can, however, be chained with more files of the same length. The pass continues to loop until the whole machine code file has been processed. As I indicated above, the first batch of source code is saved with .A appended. Subsequent batches follow the series .B, .C, and so on.

During pass five, Sourcemaster produces labels from the table of addresses collected earlier. As each label is used, its address entry in the table is set to zero. If the program counter exceeds the current label address, the label pointer is incremented and its address stays in the table.

Pass six then tidies up. It prepares and saves a library "equates" file made up of all the unused labels in the address table. The filename follows the above sequence of names, but has .@ appended. This file also serves as the master file for reassembly; all the other files are linked to it with .FIL directives.

The Art of Using Sourcemaster

Since the purpose of Sourcemaster is to produce source code you can study and edit, it is, as I mentioned, best to avoid spurious code. By spurious I mean the sort of code you see when disassembling text tables and the like. These are not intended to be treated as code areas and usually produce many lines of ???s. Try the .D monitor command at $A000 in the Basic ROM, and you'll see what I mean. Such areas can also produce what appears to be proper code but generate useless labels. Look at the disassembly at $A013–$A01B for an example.

Unassemblers start off by treating the whole machine code file as genuine code. Most do not allow user input of known noncode regions; you're expected to edit them out afterwards. However, Sourcemaster lets you enter a series of more than 60 such regions. The easiest way to detect these regions is to do a preliminary scan of the machine code using the .D feature of a monitor. Make a note of the start and finish addresses of any odd-looking code areas. As stated earlier, you don't need to be exact in this scan, as reassembly will still work properly. However, accuracy will reduce the amount of source code editing later. Usually it's not even essential that tables be designated. It all depends on how attractive you want to make the final source code file.

Sourcemaster has another advanced feature—its treatment of the Bit command. Often this command is used because it affects only the Flags register and can be allowed to run through a "hidden" entry point. The following code segment demonstrates this, with ... indicating additional code:

```
C000    JMP VALUE1

        ...
        ...
        ...
```

```
C010      LDA #$00
C012      .BYTE $2C
C013 VALUE1 LDA #$80
        ...
        ...
```

This code will disassemble as:

```
C000      JMP $C013
        ...
        ...
        ...
C010      LDA #$00
C012      BIT $80A9
        ...
        ...
```

The example shows how the Bit instruction effectively masks the true code meaning. Sourcemaster recognizes this and produces the first construction where a label corresponding to VALUE1 is found. Otherwise, it will produce the second construction. However, this feature may omit a label when applied to noncode regions. Should the problem arise for you, it's easy to fix using the Commodore editor package. You only need to add the missing label.

Once the primary source code has been produced, your real work—studying and editing the code—begins. The primary labels are formed directly from their addresses in the machine code. However, well written source code uses meaningful names for labels so that the process involved in a subroutine, called CHROUT, for example, is obvious. The editing is done with the editor in the Development Package, particularly the Change String command. You should add enough comments to the source code lines to be able to follow the program's workings. Also, convert any undetected spurious coding to byte tables.

These three processes work together. As you annotate lines and add meaningful labels, the program logic will begin to appear. As it appears, you can add comments and labels with greater confidence. Doing the three together comprises most of the art of unassembling.

The extent of your editing will depend on your reasons for requiring the source code. Many simple changes can be made with a minimum of editing, since it's often necessary to change only small sections of the machine code, or even just the code start address.

Doing Your Own Thing

The best way to acquire unassembly skill is to do it. Try working through some of your own code, to get a feel for what Sourcemaster does. Remember that you must reassemble starting with the file with .@ appended to its name. Try the unassembly with and without using the table feature. Compare the Sourcemaster output with the original source and with the .D monitor output. The disassembly and unassembly should be very similar, except the unassembly will have labels made from addresses prefixed by AD.

With a basic understanding of machine code, the unassembly process is not hard to follow. If you want to improve your own programming by studying other people's source code, just remember that there's good code and bad code, and you should learn from the former, not the latter. ■

NOTE: Remember, next issue, ReRUN will have a new, easier interface. Also, all ReRUN documentation will be contained on the disk itself. See you next time!

RERUN

EDITOR-IN-CHIEF
DENNIS BRISSON

TECHNICAL MANAGER
TIM WALSH

MANAGING EDITOR
VINOY LAUGHNER

NEW PRODUCTS EDITOR
JANICE GREAVES

COPY EDITOR
PEG LEPAGE

ART DIRECTOR
HOWARD G. HAPP

DESIGN AND LAYOUT
ANN DILLON

TYPESETTING
DEBRA DAVIES

FULFILLMENT CONSULTANT
DEBBIE BOURGAULT

9 Programs on this Disk

From the September/October RUN:

- MultiCopy
- Rollerdash
- Fraction Action
- Classy Graphics
- 128 Mode

Plus These Bonus Programs:

- Scramble
- Menu Runner
- Menu Maker
- Sourcemaster

If any manufacturing defect becomes apparent, the defective disk will be replaced free of charge if returned by prepaid mail within 30 days of purchase. Send it, with a letter specifying the defect, to:

ReRUN • 80 Elm Street • Peterborough, NH 03458

Replacements will not be made if the disk has been altered, repaired or misused through negligence, or if it shows signs of excessive wear or is damaged by equipment.

ReRUN contains programs taken directly from listings prepared to accompany articles in *RUN* Magazine, as well as bonus programs. These programs will not run under all system configurations. As your guide, use the RUN It Right information included with each article.